

## Model-driven architecture: generating models from Symfony framework

M'hamed Rahmouni<sup>1</sup>, Chaymae Talbi<sup>2</sup>, Soumia Ziti<sup>2</sup>

<sup>1</sup>Research Team of Intelligent Processing and Security of Systems (IPSS), Department of Computer Science, Ecole Normale Supérieure (ENS) in Rabat, Mohammed V University in Rabat, Rabat, Morocco

<sup>2</sup>Research Team of Intelligent Processing and Security of Systems (IPSS), Department of Computer Science, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco

### Article Info

#### Article history:

Received Jan 10, 2023

Revised Jan 14, 2023

Accepted Jan 31, 2023

#### Keywords:

Atlas transformation language

Metamodel

Model-driven architecture

MVC2

PHP Symfony framework

Platform-independent model

Platform-specific model

### ABSTRACT

The web application development industry is constantly growing due to the extensive use of web applications in different devices, most of them run on Android, iOS, and Windows Phone operating systems. However, the development of applications designed for platforms requires more concerns such as code efficiency, interaction with devices, and speed of market penetration. The model-driven approach (MDA) combined with unified modeling language (UML) could provide abstraction and automation for software developers. This paper presents an MDA approach for the development of web applications based on the Symfony framework, UML modeling, model transformation, and then automatic code generation in order to facilitate and accelerate the development of web applications. The first step of this work is to establish the metamodel of Symfony framework and the metamodel of UML class diagram. In the second step, the various transformation rules between the source and target metamodels are first defined. Atlas transformation language (ATL) implements these rules. The result of this transformation is a platform-specific model (PSM) represented by Ecore language. The generated PSM model represents the input model of model-to-code (M2C) transformation for generating the code of web applications. To validate this work, we have implemented a case study.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



### Corresponding Author:

M'hamed Rahmouni

Research Team of Intelligent Processing and Security of Systems (IPSS)

Department of Computer Science, Ecole Normale Supérieure (ENS) in Rabat

Mohammed V University in Rabat

Takaddoum, Mohammed Bel Hassan El Ouazzani Street, B.P. 5118, Rabat, Morocco

Email: mhammed.rahmouni@ens.um5.ac.ma

## 1. INTRODUCTION

Today, automatic code generation from models is one of the key elements that allow manufacturers to develop increasingly complex critical software [1]. This also makes it possible to control the production of large volumes of lines of code, within very short deadlines and at very competitive cost prices for this type of product, and also brings rigor and increases the quality level of the software. The present work, which aims to automatically generate the code of a web application, is an original work because the Symfony framework metamodel is implemented and presented for the first time in this paper. Moreover, the said paper presents an approach for generating the code of web applications based on the PHP (PHP: Hypertext Preprocessor) Symfony framework [2], [3]. This framework uses a model-view-controller (MVC) [4] architecture which allows for better teamwork and better distribution of work.

In this paper, we chose the Symfony framework in order to facilitate the development of web applications [5]. We justify our choice by the fact that Symfony is a framework powerful, secure, tailor-made, and constantly evolving. Thanks to its programming method, clear design, and readability, this framework is easy to use. The use of bundles (or groups of products) and components makes it a perfect solution for websites and applications of any size and complexity.

In order to achieve this work, we applied the model-driven architecture (MDA) approach [6]-[8] combined with unified modeling language (UML) [9] which could provide abstraction and automation for software developers. The separation of concerns is the foundation of the MDA approach. Thanks to modeling, it is able to consider the business and technical aspects of an application independently. The model, in the MDA approach, is used to represent all layers of an application, including data exchange with source systems, application objects, and their methods, machine learning algorithms [10]-[13], artificial intelligence (AI), and the user interface of the application. This approach can also take into account the development of software applications that would be deployed in the cloud, the software cloud as a service (SaaS) [14], [15].

Thus, the objective of this paper is to present a model-driven architecture (MDA) approach based on UML modeling, atlas transformation language (ATL) [16], [17] transformation, and automatic code generation for the development of web applications based on the Symfony framework in order to facilitate and accelerate generally the development of web applications. The first step of this approach is to establish the metamodels of Symfony framework and UML class diagram. The second step called model-to-model (M2M) transformation begins by establishing and defining the different traceability relationships between the source and target metamodels. These relationships are called rules in the notion of ATL transformation language and are written in this language. The transformation result of this work is a platform-specific model (PSM) depicted by Ecore language [7]. The generated PSM model represents the input model of the model-to-code (M2C) transformation for generating the web application code. To validate this work, we have implemented a case study. This last transformation, M2C, will be the subject of our next work so that this work is clear, comprehensible, and readable.

This paper's remaining section is organized as: Our proposal technique is presented in section 2. The UML class diagram metamodel is explained in section 3. The PHP Symfony framework metamodel is covered in section 4. The transformation rules implementation is the focus of section 5. The application of transformation rules is the focus of section 6. Part 7 is devoted to the primary relevant study, while section 8 concludes with recommendations for further work.

## 2. OUR PROPOSAL

In this paper, we will shed light on our disciplined proposed method which is intended to establish the metamodel of Symfony framework and thereafter transform it to the PSM model and then generate automatically from this PSM model the web application code, respecting the MDA approach. Thus, in this paper we use a structured and pedagogically well-founded method, which consists in first defining the computational-Independent Model (CIM) and platform-specific model (PIM) of the application through a UML model, then automatically transforming these PIMs into PSMs then, if necessary, adding the missing elements to these PSMs to complete them (refinement). The next step concerns code generation, in which we prepare the different templates suitable for the generated PSM in view to automatically generate the requested code. It is important to note that in this work, the CIM represents the case study specification, the PIM represents the metamodel of UML class diagram while the PSM represents the metamodel of Symfony framework. The cited MDA approach is automated and therefore productive. However, it allows for significant gains in productivity.

To carry out this work, we used Eclipse [18] while exploiting the possibility offered by it which resides in the plugins such as the plugin of ATL language in order to implement the PIM into PSM transformation rules, Ecore for representing the different models, and Acceleo [19] in view to implement the PSM into Code transformation. In this paper, we generate only the PSM model which will subsequently be the input model for ACCELEO in order to generate the web applications code. This step will be the objective of future work in order not to fill the present work and subsequently to leave it clear and understandable.

## 3. UML CLASS DIAGRAM (UMLCD) METAMODEL

UML is recommended by the MDA approach as being the language to be used to carry out analysis and design models independent of the implementation platforms. This is why in the MDA vocabulary these models are called platform-independent model (PIM) [6]. Figure 1 shows the different metaclasses that make up the PIM source metamodel (UML Class Diagram). These metaclasses are as shown as follow:

- UMLPackage: It symbolizes the UML package notion.
- Class: Explains the notion of Class in the UML context.
- Classifier: Represents the UML class concept and the data type concept.
- Attribute: Represents the notion of UML class properties or references.
- Operation: Represents the concept of method in the UML context.
- Parameter: Represents the attributes and parameters of the method.
- Primitive data type: Represents the notion of primitive type. Like as: boolean, byte, char, short, int, long, float, and double.
- Association: Explains the notion of association in the UML context. This is a relationship between two more metaclasses.

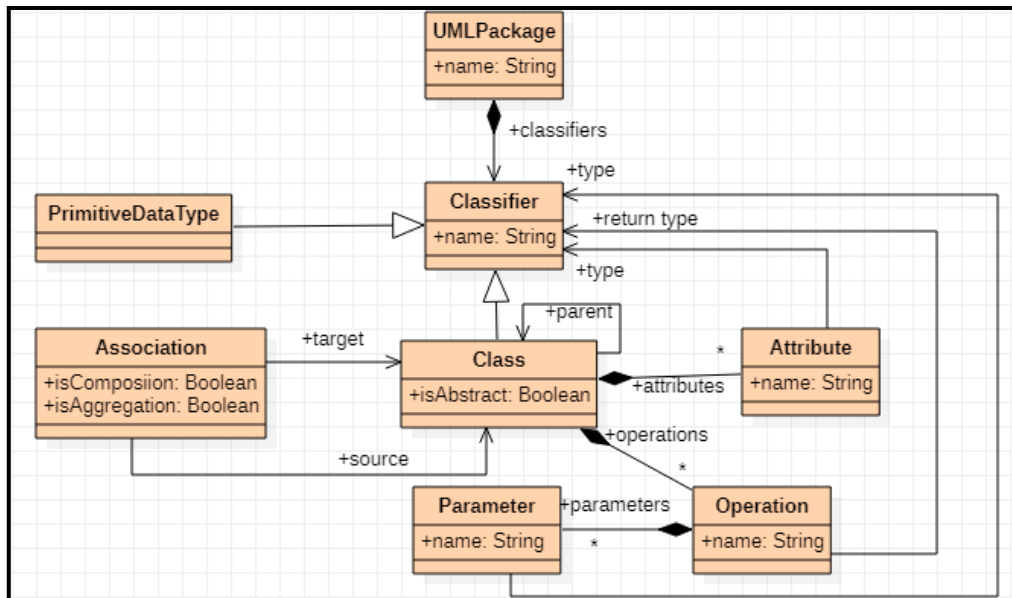


Figure 1. An excerpt of UML class diagram metamodel

#### 4. SYMFONY FRAMEWORK METAMODEL

Symfony is a comprehensive framework created to streamline the creation of web applications with a number of essential features [2], [3]. Thus, it divides the display views, server logic, and business rules of a web application. It includes a lot of tools and classes designed to accelerate the creation of complicated web applications. Additionally, it automates routine operations so the developer may concentrate entirely on the details of an application. Because of these advantages, it is no longer necessary to create a brand-new web application every time one is created.

To exploit the Symfony framework in the field of model-driven architecture (MDA), we will model this framework in order to establish its metamodel. The latter is composed of a set of elements that are represented by a UML class diagram. Figure 2 presents the metamodel of the Symfony framework. The different metaclasses that make up the Symfony framework target metamodel and their notions are listed as follow:

- Symfony package: Represents the notion of a package for the Symfony framework.
- Models: Refers to the set of models in the model 2 view controller (MVC2) architectural concept.
- Model: In the context of the MVC2 pattern, it represents the idea of the Model class.
- Controllers: In the context of MVC2 architecture, represents the idea of a package controller.
- Abstract controller: It is an optional base controller. It can be extended to access helper methods.
- Controller: The Controller class in the MVC2 architectural concept.
- Views: This is the view package that regroups the different Template classes in the notion of MVC2 architecture.
- Template: Represents the concept of the view in the notion of MVC2 architecture. It allows separating the HTML code from the PHP code.
- Actions: Depicts the functions package notion.

- Function: Explains the Action class concept in the notion of PHP Symfony framework.
- Forms: This term refers to the collection of Form classes.
- Form: Explains the Form class concept. It models the concept of form in the notion of MVC2 pattern.
- HttpRequest: This is the concept of HttpServletRequest class.
- HttpResponse: Represents the HttpServletResponse class notion.
- Router: Allows directing a URL (or a pattern of URLs) to a controller method called Action.

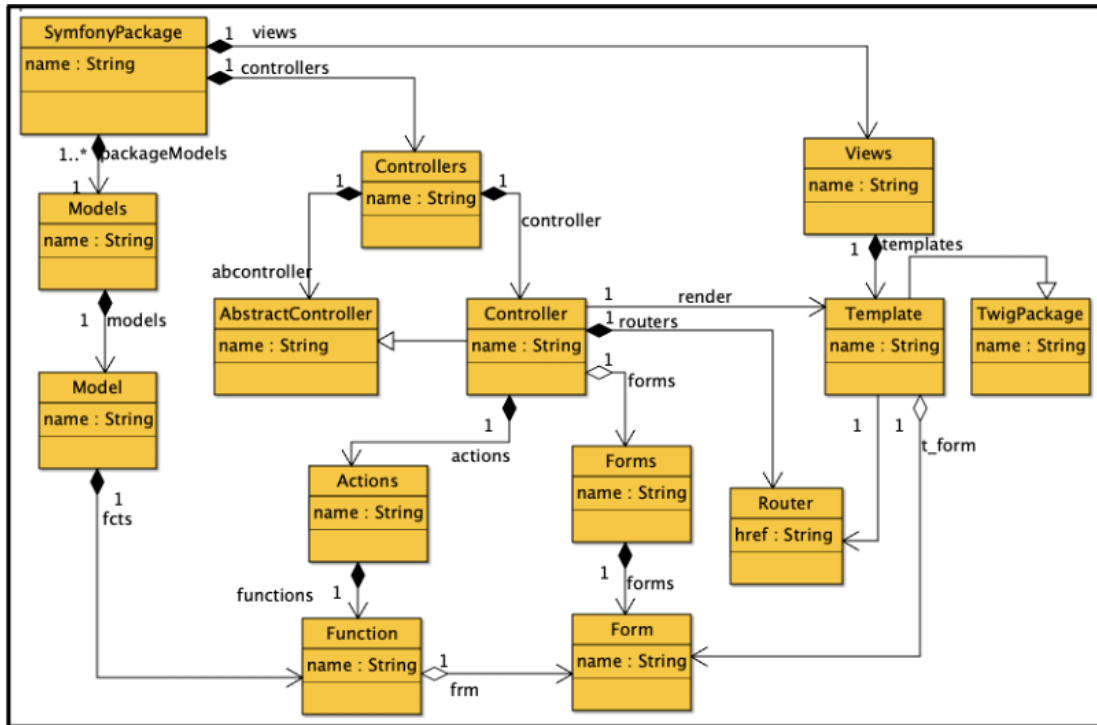


Figure 2. Symfony framework metamodel

**5. TRANSFORMATION RULES IMPLEMENTATION**

In the ATL language concept, the generation of the target model is carried out thanks to the transformation rule specifications. A transformation rule makes it possible to match certain elements of the source model and to generate from these elements certain elements of the target model [6]. In this section, we present the different traceability relationships between the metaclasses of source and target metamodels. Each traceability relationship represents a transformation rule implemented by the ATL transformation language [17].

**5.1. Specification rules between source and target metamodel**

Rules in model transformations handle the mapping between source and target metamodel components. In this section, we present the different traceability relationships between the different metaclasses that make up the UML class diagram (UMLCD) and PHP Symfony metamodels. These are the transformation rules which permit to transform a UMLCD into a PHP Symfony model. These rules are listed and presented in Table 1.

Table 1. The different transformation rules from UMLCD to Symfony framework

Rule	Description
C1	Symfony package may be generated from any UML package.
C2	collection of controllers, models, and views make up the Symfony package.
C3	There are several actions and forms packages that make up the controller package.
C4	There are several view classes in the view package.
C5	There are several function classes in the actions package.
C6	One can create a model for each class.
C7	Each operation class has the ability to produce a function, form, and view.

### 5.2. Transformation rules implementation

The following section outlines the many rules used to transform the UMLCD model into the PHP Symfony model. The ATL language is used to express these rules. The fundamental rules are expressed in ATL language and are shown in Figure 3.

<pre> rule UMLPackage2SymfonyPackage{   from     a : UML!UML   to     out : Symfony!SymfonyPackage(       controllers &lt;- Sequence{cc},       models &lt;- Sequence{mm},       views &lt;- Sequence{vv}     ),     cc : Symfony!Controllers(       controller &lt;- c     ),     c : Symfony!Controller(       action &lt;- Sequence{act},       form &lt;- Sequence{fo}     ),     act : Symfony!Actions(       actiones &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm'))       },     fo : Symfony!Forms(       formes &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm3'))       },     mm : Symfony!Models(       model &lt;- Sequence{thisModule.allMethodDefClass         -&gt;collect(e   thisModule.resolveTemp(e, 'frm1'))       },     vv : Symfony!Views(       template &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm2'))       )   } </pre>	<pre> rule Operation2TemplateFunction{   from     c : UML!Operation   to     frm : Symfony!Function(       name&lt;- c.className+'Action'     ),     frm3 : Symfony!Form(       name&lt;- c.className+'Form'     ),     frm2 : Symfony!Template(       name&lt;- c.className+'Template'+'.html'+'.twig'     ) } rule Class2Model{   from     c : UML!Class   to     frm1 : Symfony!Model (       name&lt;- c.name+'Model'     ) } </pre>
---	--

Figure 3. The main ATL transformation rules that transform the UMLCD into PHP Symfony

## 6. TRANSFORMATION RULES EXECUTION

In order to establish the link between the components of the source and destination metamodels, the execution procedure for transformation rules entails browsing through the various transformation rules. We give a case study that is a UML class diagram of a bookstore management (BSM) system in order to demonstrate and support our proposition. An Ecore model is the end outcome of this transformation. This model is known as the PSM model.

### 6.1. Case study

Bookstore management (BSM) is a concrete and sufficiently representative example of e-commerce projects, for this we want to create software for bookstore management. The fundamental objective of the future software is to allow customers to search for books by theme or topic, title, and ISBN, to build a shopping cart, then to be able to order and pay for them directly. This case study is represented by a UML class diagram. Figure 4 depicts the aforementioned BSM system UMLCD.

### 6.2. ATL transformation result

The generated PSM model shown in Figure 5 in *appendix* is made up of three essential packages which stand for the MVC acronym (Model-View-Controller). The first package stands for the package of controllers, the second for the package of views, and the third for the package of models. The controllers package consists of a set of actions and forms. The package entitled Forms represents the forms package, which contains all the forms that we will need to complete this application, and the Actions package is made up of a group of functions that represent the processing activities. In the same way, the views package which represents the Views of the application in turn also contains all the views that we will need to achieve our application. Finally, the models package which represents the package of models also contains all the models necessary for the realization of the business service management (BSM) application.

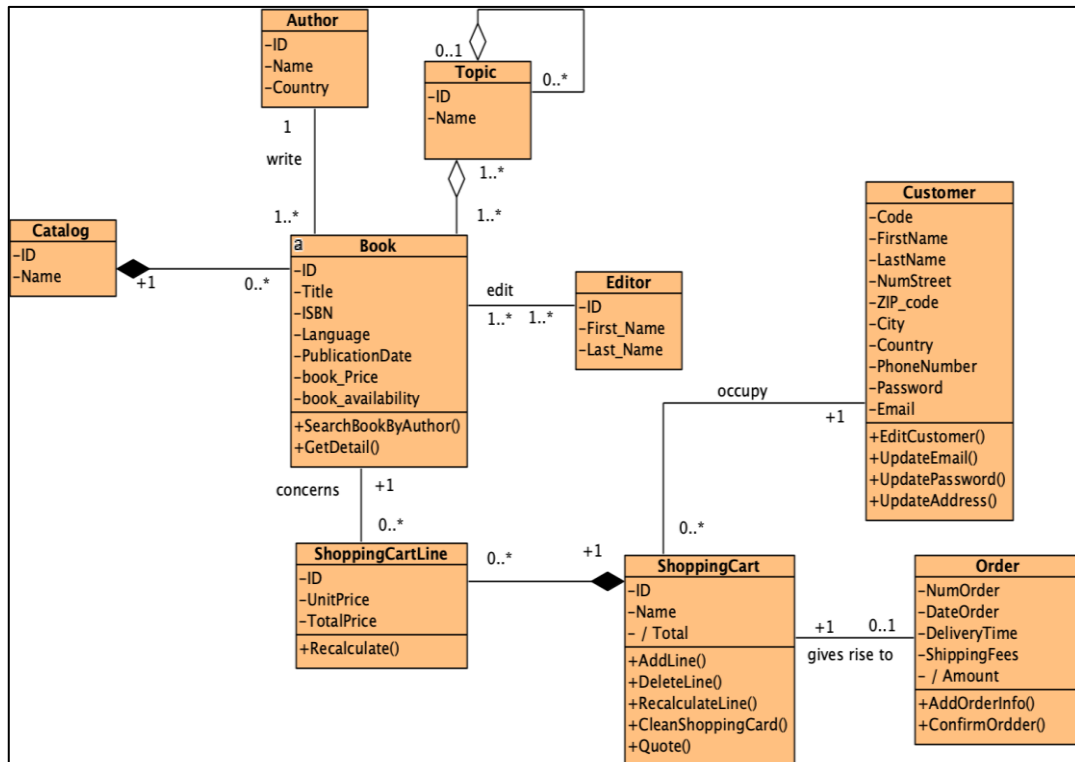


Figure 4. UMLCD of a bookstore management system.

## 7. RELATED WORK

Despite the lack of in-depth research on model-driven architecture and M2M transformation applications, M2M transformation and other contributions to the field are still contentious issues in many resources. The outcomes of the analysis covering the most relevant subset of M2M transformations to our research are provided in the section on related work. According to Rhazali *et al.* [20], M2M is being transformed from the level of a CIM to a PIM. An approach to converting service-oriented business concepts into web-based design models is provided by the author. The topic of CIM conversion to PIM is covered in the second work [20]. However, Melouk *et al.* [21] propose a new method for transforming the CIM into the PIM and then the PSM model. The MDA approach is adhered to by this methodology. In this study, PIM levels are represented by UML models, whereas PSM levels are represented using interaction flow modeling language (IFML), an OMG standard for building web interfaces. On the other hand, the CIM level is described using the object management group (OMG) standard for modeling business processes, the business process model and notation, or BPMN.

The case study of course management in a department is provided by Sajji *et al.* [22] and is represented by a class diagram. The code for the classes is built by the authors using the enterprise architect application. The authors of [22] propose a method that meets with the guidelines provided by IFML and MDA. This method uses MDA model transformations to advance through the PSM model from a business process model at the CIM level to automatically generate graphical user interfaces (GUI).

STS4IoT, a tool for model-driven internet of things (IoT) that automatically generates code and UML profiles, is introduced by Plazas *et al.* [23]. STS4IoT makes it possible to develop and implement an IoT application using only the information that is required by bridging the gap between the worlds of IoT and database design. IoT data is created via both multiple network transformations and the mixing of streams from numerous sources. It also follows MDA guidelines to provide abstraction layers that are customized to the numerous responsibilities played during the development of the application.

In the work of Alulema *et al.* [24], IoT data may be included into RESTful cloud services using a meta-model provided. The connection between various IoT items and centralized systems, as well as their interoperability, is emphasized. They do this by illustrating and putting into action a variety of network solutions. The data composition or transformation, however, is not taken into account.

In another work, the N-tiers web model is created by Rahmouni and Mbarki [25], who integrate the Struts2, Spring IoC, and Hibernate DAO frameworks. The authors originally constructed the metamodels of

the aforementioned frameworks in this study. The PIM model is represented by the built-in metamodels of this framework. A UML class diagram serves as the PIM model for this piece of work [25]. Every element required to build an N-tiers web application is included in the PSM model that was developed.

English2OCL (En2OCL) by Salemi *et al.* [26] provides a paradigm for automatically transforming system constraints produced by English words to OCL requirements. The proposed model is based on the MDA approach. After the Maude model checker confirmed the proposed model's attributes, the authors developed the English2OCL application.

For model transformations, Saqib and Azzoni [27] provide a method for test case prioritization (TCP). The goal of this method is to identify a test case ordering that enhances fault detection efficiency. A transformation language carries out this strategy. With the help of this application transformation, test case orderings may be created from various case studies.

To make the usage of MVC-based frameworks in web application development easier, Ahmad *et al.* [28] introduce the model-driven framework method. This work's technique is based on the usage of the UML profile model and a model-to-text translation in order to automatically generate the implementation of a web application in three preset MVC-based frameworks. For a continuous cycle of software engineering in systems based on microservices, Cortellessa *et al.* [29] 's MDA technique is suggested. The authors of the current paper concentrated on spring boot and spring cloud-developed microservices applications that were deployed on Docker.

Nevertheless, in the work described by Arrhioui *et al.* [30], the authors arrive at generating the create, read, update, and delete (CRUD) apps based on the codeIgniter PHP framework. In this study, the authors first model the codeIgniter PHP framework before creating the PIM model that goes along with the framework. Finally, they set up the various Acceleo templates to produce the PHP code for the codeIgniter-based CRUD apps.

Therefore, the purpose of our paper is to use the MDA approach-based-ATL language to generate web Models of the MVC2 pattern from the PHP Symfony framework. Although the PIM and PSM metamodels differ, this study uses the same MDA technique for the portion of the PIM to PSM transformation as the studies listed above. Additionally, the PSM metamodel is introduced in this work for the first time. The result of this work is very satisfying because we managed to generate all the elements necessary for the development of an e-commerce web application from any case study.

## 8. CONCLUSION

The web application development industry is constantly growing due to the extensive use of web applications in different devices, most of them run on Android, iOS, and Windows Phone operating systems. However, the development of applications designed for platforms requires more concerns such as code efficiency, interaction with devices, and speed of market penetration. The model-driven approach (MDA) combined with unified modeling language (UML) could provide abstraction and automation for software developers. This paper presents an MDA approach for the development of web applications based on the Symfony framework, UML modeling, and model transformation in order to facilitate and accelerate the development of web applications. The first step of this work is to establish the metamodel of Symfony framework and the metamodel of UML class diagram. The second step begins by establishing the different traceability relationships between the different meta-classes that make up the UMLCD and PHP Symfony metamodels then defining the different transformation rules between the source and target metamodels. These rules are implemented by ATL transformation language. The result of this transformation is a PSM model represented by Ecore language. The generated PSM model represents the input model of model-to-code (M2C) transformation for generating the code of the web application. To validate this work, we have implemented a case study. This work's main goal is to use the PHP Symfony framework to automatically create a PHP application. To achieve this goal, we used the Ecore PSM model generated by the M2M transformation. The said model is used as an input file for a code-generating program that was created for this task using the Acceleo generator. Thus, we have successfully completed our task, which in this case was to create the online PSM model using the PHP Symfony framework. Additionally, the mentioned PSM model is entirely generated because it has all components needed to create a PHP application. In future work, we intend to use the created PSM model to automatically build a web application using the Symfony PHP framework. We are trying to create additional PSM models from other PIM case studies to validate this application. Along the same lines, we attempt to model additional frameworks such as PHP Zend to generate alternative PSM models using an MDA-based ATL transformation approach. Then, we compare the different attributes of the resulting models and their uses. The metrics used for this comparison will be clear. Thus, the requested objective is to evaluate the maturity of each framework by comparing it to the others. As a result, we are able to select and suggest the best framework to create a solid, flawless, bug-free, and end-to-end PHP web application among them.

APPENDIX






Figure 5. The generated PSM model of UMLCD2PHPSymfony



## REFERENCES

- [1] J. Ding, J. Lu, G. Wang, J. Ma, D. Kiritsis, and Y. Yan, "Code generation approach supporting complex system modeling based on graph pattern matching," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 3004–3009, 2022, doi: 10.1016/j.ifacol.2022.10.189.
- [2] Symfony, "Symfony Homepage, Symfony components." symfony.com. <https://symfony.com/components> (accessed Jan 1, 2023).
- [3] F. Potencier, *Symfony 5 : the fast track*. SENSIO SA, 2020.
- [4] A. Fathonih, D. S. Maylawati, and M. Ali Ramdhani, "Model-view-controller approach for e-Zakah," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 15, no. 2, p. 1054, Aug. 2019, doi: 10.11591/ijeecs.v15.i2.pp1054-1065.
- [5] S. Sherin, A. Muqet, M. U. Khan, and M. Z. Iqbal, "QExplore: An exploration strategy for dynamic web applications using guided search," *Journal of Systems and Software*, vol. 195, 2023, doi: 10.1016/j.jss.2022.111512.
- [6] G. Sebastián, J. A. Gallud, and R. Tesoriero, "Code generation using model driven architecture: A systematic mapping study," *Journal of Computer Languages*, vol. 56, p. 100935, Feb. 2020, doi: 10.1016/j.cola.2019.100935.
- [7] X. Blanc and O. Salvatori, "MDA en action: Ingénierie logicielle guidée par les modèles," *Eyrolles*, p. 266, 2011.
- [8] N. Kharmoum, K. El Bouchti, N. Laaz, W. Rhalem, and Y. Rhazali, "Transformations' study between requirements models and business process models in MDA approach," *Procedia Computer Science*, vol. 170, pp. 819–824, 2020, doi: 10.1016/j.procs.2020.03.150.
- [9] F. Wang, "UML diagram classification model based on convolution neural network," *Optik*, p. 170463, 2022, doi: 10.1016/j.ijleo.2022.170463.
- [10] Y.-S. Ren, C.-Q. Ma, X.-L. Kong, K. Baltas, and Q. Zureigat, "Past, present, and future of the application of machine learning in cryptocurrency research," *Research in International Business and Finance*, vol. 63, p. 101799, Dec. 2022, doi: 10.1016/j.ribaf.2022.101799.
- [11] M. Knott, F. Perez-Cruz, and T. Defraeye, "Facilitated machine learning for image-based fruit quality assessment," *Journal of Food Engineering*, vol. 345, p. 111401, May 2023, doi: 10.1016/j.jfoodeng.2022.111401.
- [12] M. Kaur and D. Rattan, "A systematic literature review on the use of machine learning in code clone research," *Computer Science Review*, vol. 47, p. 100528, Feb. 2023, doi: 10.1016/j.cosrev.2022.100528.
- [13] M. Hobensack, J. Song, D. Scharp, K. H. Bowles, and M. Topaz, "Machine learning applied to electronic health record data in home healthcare: A scoping review," *International Journal of Medical Informatics*, vol. 170, p. 104978, Feb. 2023, doi: 10.1016/j.ijmedinf.2022.104978.
- [14] M. F. Falah *et al.*, "Comparison of cloud computing providers for development of big data and internet of things application," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 3, pp. 1723–1730, 2021, doi: 10.11591/ijeecs.v22.i3.pp1723-1730.
- [15] A. Adesina, A. Adebisi, and C. Ayo, "Detection and extraction of digital footprints from the iDrive cloud storage using web browser forensics analysis," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 26, no. 1, p. 550, Apr. 2022, doi: 10.11591/ijeecs.v26.i1.pp550-559.
- [16] B. Alkhazi, C. Abid, M. Kessentini, and M. Wimmer, "On the value of quality attributes for refactoring ATL model transformations: A multi-objective approach," *Information and Software Technology*, vol. 120, p. 106243, Apr. 2020, doi: 10.1016/j.infsof.2019.106243.
- [17] "ATL Language." <https://www.eclipse.org/atl/> (accessed Jan 1, 2023).
- [18] Eclipse, "{E}clipse {F}oundation," *{E}clipse {F}oundation*, 2008. <http://www.eclipse.org/> (accessed Jan 1, 2023).
- [19] Acceleo Homepage, "Acceleo." <https://www.eclipse.org/acceleo> (accessed Jan 1, 2023).
- [20] Y. Rhazali, Y. Hadi, and A. Mouloudi, "CIM to PIM transformation in MDA: from service-oriented business models to web-based design models," *International Journal of Software Engineering and Its Applications*, vol. 10, no. 4, pp. 125–142, Apr. 2016, doi: 10.14257/ijseia.2016.10.4.13.
- [21] M. Melouk, Y. Rhazali, and H. Youssef, "An approach for transforming CIM to PIM up To PSM in MDA," *Procedia Computer Science*, vol. 170, pp. 869–874, 2020, doi: 10.1016/j.procs.2020.03.122.
- [22] A. Sajji, Y. Rhazali, and Y. Hadi, "An approach to automate generation of graphical user interfaces through IFML," *Procedia Computer Science*, vol. 201, no. C, pp. 621–626, 2022, doi: 10.1016/j.procs.2022.03.081.
- [23] J. E. Plazas *et al.*, "Sense, transform & send for the internet of things (STS4IoT): UML profile for data-centric IoT applications," *Data & Knowledge Engineering*, vol. 139, p. 101971, May 2022, doi: 10.1016/j.datak.2021.101971.
- [24] D. Alulema, J. Criado, L. Iribarne, A. J. Fernández-García, and R. Ayala, "A model-driven engineering approach for the service integration of IoT systems," *Cluster Computing*, vol. 23, no. 3, pp. 1937–1954, Sep. 2020, doi: 10.1007/s10586-020-03150-x.
- [25] M. Rahmouni and S. Mbarki, "MDA-based modeling and transformation to generate N-Tiers web models," *Journal of Software*, vol. 10, no. 3, pp. 222–238, Mar. 2015, doi: 10.17706/jsw.10.3.222-238.
- [26] S. Salemi, A. Selamat, and M. Penhaker, "A model transformation framework to increase OCL usability," *Journal of King Saud University - Computer and Information Sciences*, vol. 28, no. 1, pp. 13–26, Jan. 2016, doi: 10.1016/j.jksuci.2015.04.002.
- [27] S. Iqbal and I. Al-Azzoni, "Test case prioritization for model transformations," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6324–6338, Sep. 2022, doi: 10.1016/j.jksuci.2021.08.011.
- [28] S. I. Ahmad, T. Rana, and A. Maqbool, "A model-driven framework for the development of MVC-based (Web) application," *Arabian Journal for Science and Engineering*, vol. 47, no. 2, pp. 1733–1747, Feb. 2022, doi: 10.1007/s13369-021-06087-4.
- [29] V. Cortellessa, D. Di Pompeo, R. Eramo, and M. Tucci, "A model-driven approach for continuous performance engineering in microservice-based systems," *Journal of Systems and Software*, vol. 183, p. 111084, Jan. 2022, doi: 10.1016/j.jss.2021.111084.
- [30] K. Arrhioui, S. Mbarki, O. Betari, and M. Erramdani, "A model driven approach for modeling and generating PHP codeigniter based applications," *Transactions on Machine Learning and Artificial Intelligence*, vol. 5, no. 4, Aug. 2017, doi: 10.14738/tmlai.54.3189.




**BIOGRAPHIES OF AUTHORS**

**M'hamed Rahmouni**    received his diploma of higher approfondie studies in computer science and telecommunication from the Faculty of Science, Ibn Tofail University, Kenitra, Morocco, in 2007, and doctorat of high graduate studies degrees in computer sciences from Ibn Tofail University, Kenitra, Morocco, in 2015. In 2018 he is attached to the Higher Normal School (ENS-Rabat) Mohammed V University in Rabat, Morocco where he is currently a professor in Department of computer science. He participated in various international congresses in MDA (Model Driven Architecture) integrating new technologies XML, EJB, MVC, and Web Services. and he published many papers in the MDA domain. His research interests include software engineering, model driven architecture, software tests, smart cities. He can be contacted at email: md.rahmouni@yahoo.fr.



**Chaymae Talbi**    received her Master degree in telecommunications systems from the Faculty of Science, Ibn Tofail University in KENITRA, Morocco, in 2021. Her research interests include the contribution of 5G technology to smart cities. Currently, she is a telecommunication engineer in a subcontracting company of a French operator. She can be contacted at email: talbichaymae@gmail.com.



**Soumia Ziti**    is a Franco-Moroccan teacher-researcher lady at the Faculty of Sciences of Mohammed V University in Rabat since 2007. She holds a Ph.D. in computer science in the field of graph theory, and a diploma in advanced studies in fundamental computer science, both obtained at the University of Orleans in France. She is a laureate of a Baccalaureate in Mathematical Sciences and she obtained a diploma of general university studies in mathematics and physics, mathematics option and a master's degree in science and technology in computer science at Hassan II University in Morocco. His areas of expertise and research are graph theory, information systems, artificial intelligence, data science, software development engineering, modeling of relational databases and big data, cryptography and numerical methods and simulations in spintronics. She has over than sixty publications in high-level international journals and conferences in several research areas. In addition, she coordinates or participates in several educational or socio-economic projects. She can be contacted at email: s.ziti@um5r.ac.ma.